TUPLES

python

Tuple Introduction

➢A tuple is an ordered sequence of elements of different data types, such as integer, float, string, list or even a tuple.

➢ Elements of a tuple are enclosed in parenthesis (round brackets) and are separated by commas.

➢Like list and string, elements of a tuple can be accessed using index values, starting from 0.

```
>>> tuple = ()                                      # Empty Tuple
>>> Tuple = (1)                                     # Tuple with single element
>>> tuple1 = (1,2,3,4,5)                            # tuple of integers
>>> tuple2 =('Economics',87,'Accountancy',89.6)     # tuple of mixed data types
```

NOTE:

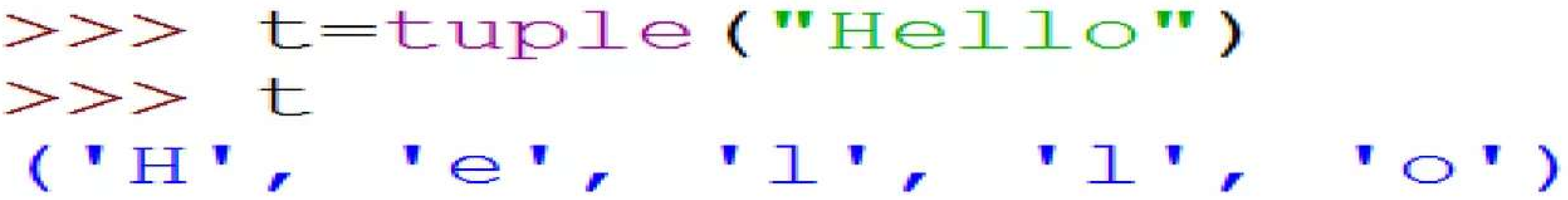
If we assign the value without comma it is treated as integer.

It should be noted that a sequence without parenthesis is treated as tuple by default.

Creation of Tuple

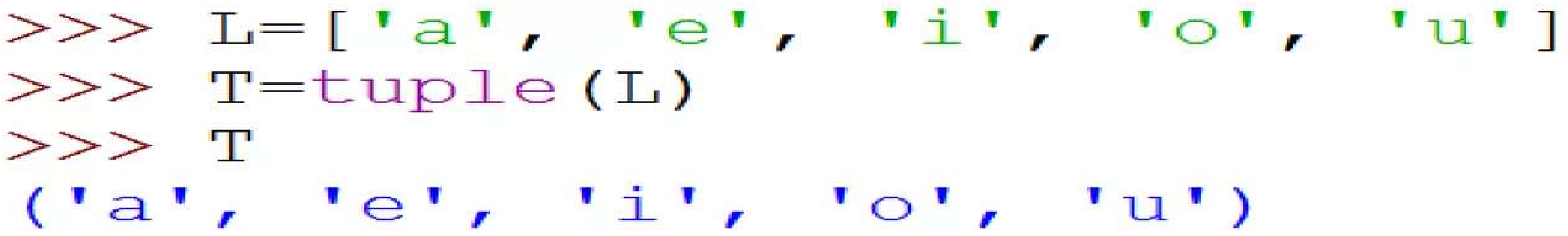
tuple() function is used to create a tuple from other sequences.

Tuple Creation from List :



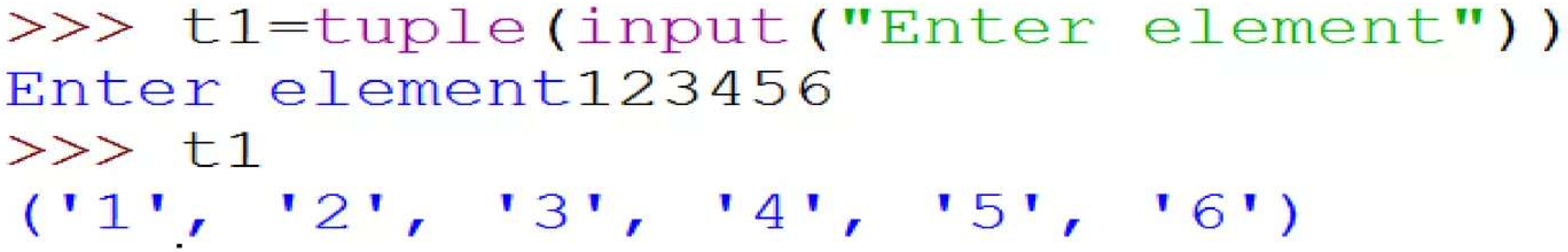
```
>>> t=tuple("Hello")
>>> t
('H', 'e', 'l', 'l', 'o')
```

Tuple creation from String :



```
>>> L=['a', 'e', 'i', 'o', 'u']
>>> T=tuple(L)
>>> T
('a', 'e', 'i', 'o', 'u')
```

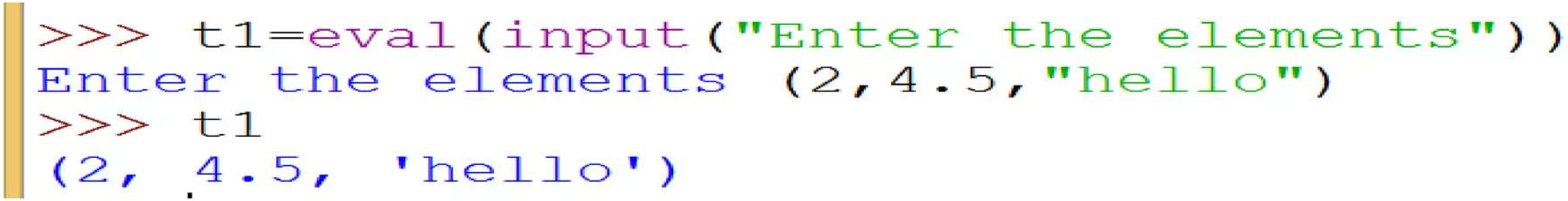
Tuple creation from input()



```
>>> t1=tuple(input("Enter element"))
Enter element123456
>>> t1
('1', '2', '3', '4', '5', '6')
```

Tuple creation using eval() :

for Ex: Tuple = eval(input("Enter elements"))



```
>>> t1=eval(input("Enter the elements"))
Enter the elements (2,4.5,"hello")
>>> t1
(2, 4.5, 'hello')
```

Accessing Elements in a Tuple

Elements of a tuple can be accessed in the same way as a list or string using indexing and slicing.

```
>>> tuple1 = (2,4,6,8,10,12)        # returns the first element of tuple1
>>> tuple1[0]                        # returns fourth element of tuple1
2
>>> tuple1[3]
8
>>> tuple1[15]                      # returns error as index is out of range
      IndexError: tuple index out of range index

>>> tuple1[1+4]                     # an expression resulting in an integer
12
>>> tuple1[-1]                      # returns first element from right
12
```

NOTE:

Tuple is an immutable data type. It means that the elements of a tuple cannot be changed

Tuple Operations

Concatenation It allows to join tuples using concatenation operator depicted by symbol +. We can also create a new tuple which contains the result of this concatenation operation.

```
>>> tuple1 = (1,3,5,7,9)
>>> tuple2 = (2,4,6,8,10)
 >>> tuple1 + tuple2              # concatenates two tuples (1, 3, 5, 7, 9, 2, 4, 6, 8, 10)
```

Concatenation operator (+) can also be used for extending an existing tuple. When we extend a tuple using concatenation a new tuple is created.

```
>>> tuple3(1, 2, 3, 4, 5, 6)                    # more than one elements are appended
>>> tuple4 = tuple3 + (7,8,9)
>>> tuple4
(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

Repetition It is denoted by the symbol *(asterisk).

It is used to repeat elements of a tuple. We can repeat the tuple elements. The repetition operator requires the first operand to be a tuple and the second operand to be an integer only.

```
>>> tuple1 = ('Hello','World')
>>> tuple1 * 2                                  #tuple with single element
 ('Hello', 'World', 'Hello', 'World’)
```

Membership

The in operator checks if the element is present in the tuple and returns True, else it returns False.

```
>>> tuple1 = ('Red','Green','Blue')
>>> 'Green' in tuple1
      True
```

The not in operator returns True if the element is not present in the tuple, else it returns False.

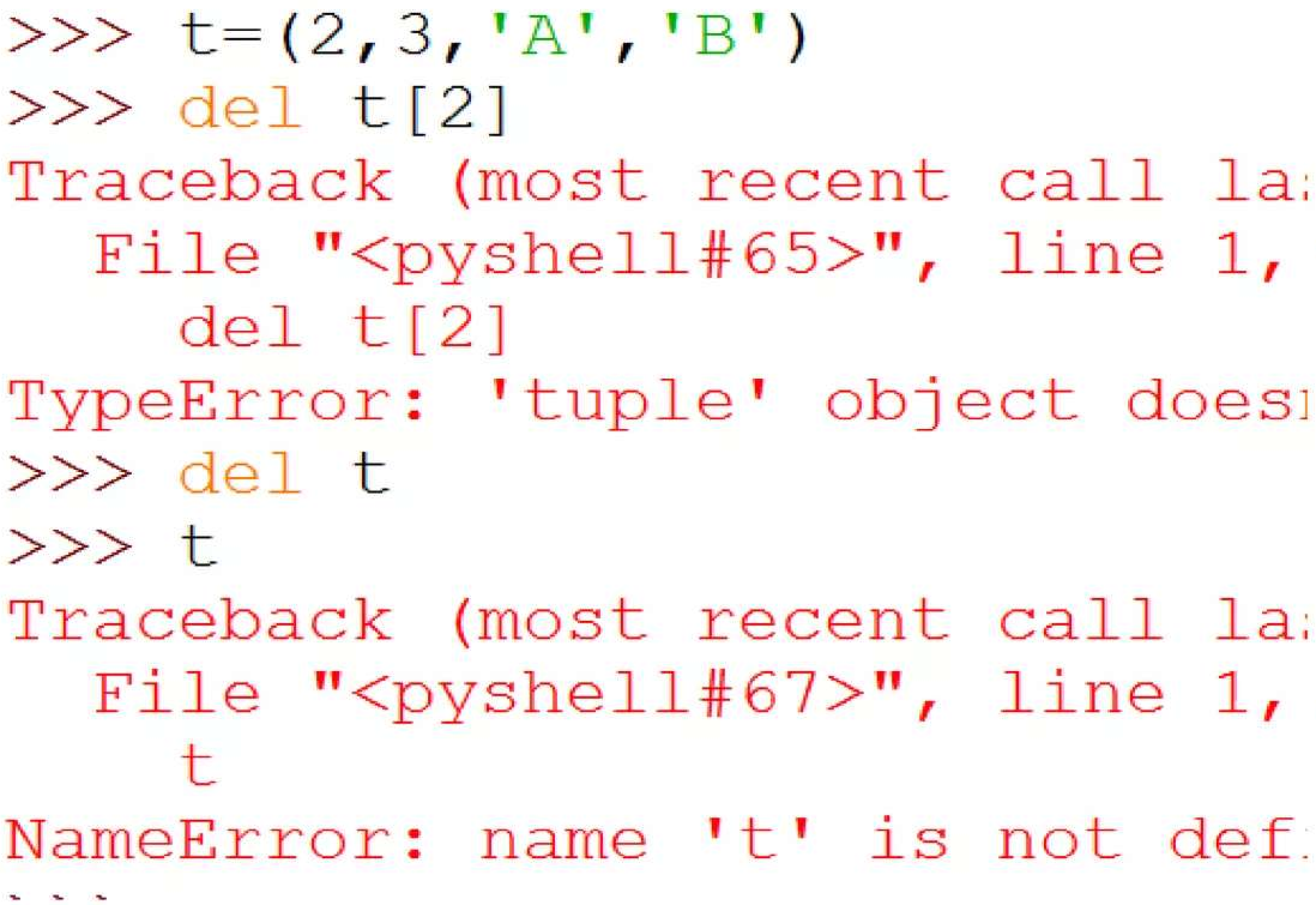
```
>>> tuple1 = ('Red','Green','Blue')
>>> 'Green' not in tuple1
      False
```

Slicing

Like string and list, slicing can be applied to tuples also.

```
>>> tuple1 = (10,20,30,40,50,60,70,80)      # tuple1 is a tuple
 >>> tuple1[2:7] (30, 40, 50, 60, 70)        # elements from index 2 to index 6
>>> tuple1[0:len(tuple1)]                   # all elements of tuple are printed
(10, 20, 30, 40, 50, 60, 70, 80)
>>> tuple1[:5] (10, 20, 30, 40, 50)         # slice starts from zero index
>>> tuple1[2:] (30, 40, 50, 60, 70, 80)     # slice is till end of the tuple
```

Tuple Deletion

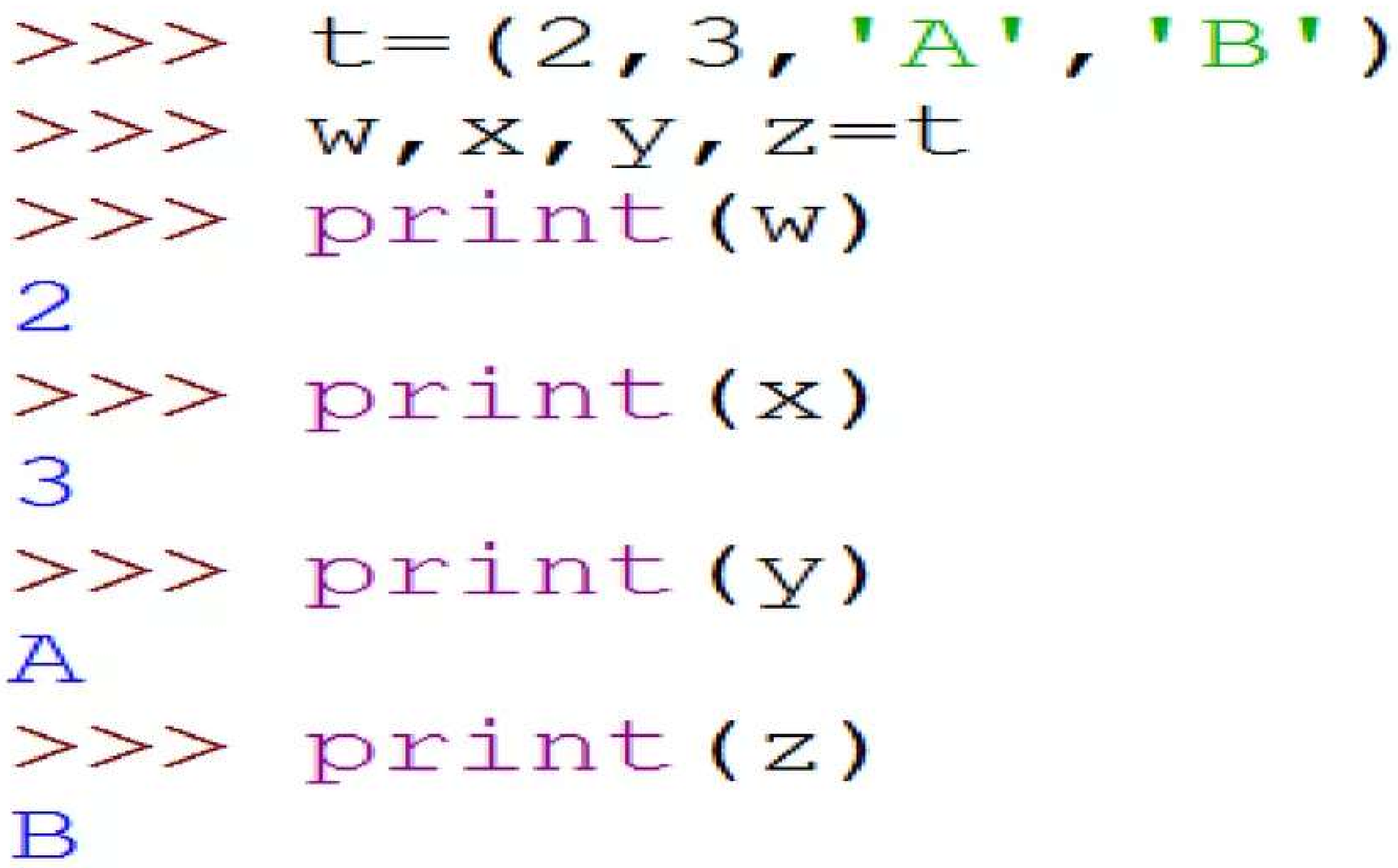


```
>>> t=(2,3,'A','B')
>>> del t[2]
Traceback (most recent call la
  File "<pyshell#65>", line 1,
    del t[2]
TypeError: 'tuple' object does
>>> del t
>>> t
Traceback (most recent call la
  File "<pyshell#67>", line 1,
    t
NameError: name 't' is not def
```

Error shown because deletion of a single element is also possible.

Complete tuple has been deleted. Now error shown on printing of tuple.

Tuple unpacking



```
>>> t=(2,3,'A','B')
>>> w,x,y,z=t
>>> print(w)
2
>>> print(x)
3
>>> print(y)
A
>>> print(z)
B
```

Tuple Methods and Built-in Functions

Method	Description	Example

len()	Returns the length or the number of elements of the tuple passed as the argument	>>> tuple1 = (10,20,30,40,50)
		>>> len(tuple1)
		5

tuple()	Creates an empty tuple if no argument is passed
	Creates a tuple if a sequence is passed as argument
		>>> tuple1 = tuple()
		>>> tuple1 ( )
		>>> tuple1 = tuple('aeiou') **#string**
		>>> tuple1 ('a', 'e', 'i', 'o', 'u')
		>>> tuple2 = tuple([1,2,3]) **#list**
		>>> tuple2 (1, 2, 3)
		>>> tuple3 = tuple(range(5))
		>>> tuple3 (0, 1, 2, 3, 4)

count()	Returns the number of times the given element appears in the tuple	>>> tuple1 = (10,20,30,10,40,10,50)
		>>> tuple1.count(10)
		3
		>>> tuple1.count(90)
		0

Method	Description	Example

index()	Returns the index of the first occurrence of the element in the given tuple	>>> tuple1 = (10,20,30,40,50)
		>>> tuple1.index(30)
		2
		>>> tuple1.index(90)
		ValueError: tuple.index(x): x not in tuple

sorted()	Takes elements in the tuple and returns a new sorted list. It should be noted that, sorted() does not make any change to the original tuple	>>> tuple1 = ("Rama","Heena","Raj", "Mohsin","Aditya")
		>>> sorted(tuple1)
		['Aditya', 'Heena', 'Mohsin', 'Raj', 'Rama']

min()	Returns minimum or smallest element of the tuple	>>> tuple1 = (19,12,56,18,9,87,34)
		>>> min(tuple1)
		9

max()	Returns maximum or largest element of the tuple	>>> max(tuple1) 87
sum()	Returns sum of the elements of the tuple	>>> sum(tuple1) 235

Nested Tuples

- A tuple inside another tuple is called a nested tuple.
- In the given program, roll number, name and marks (in percentage) of students are saved in a tuple.
- To store details of many such students we can create a nested tuple

#Create a nested tuple to store roll number, name and marks of students
To store records of students in tuple and print them

```
st=((101,"Aman",98),(102,"Geet",95),(103,"Sahil",87),(104,"Pawan",79))
print("S_No"," Roll_No"," Name"," Marks")
for i in range(0,len(st)):
            print((i+1),'\t',st[i][0],'\t',st[i][1],'\t',st[i][2])
```

Output:

```
S_No    Roll_No    Name     Marks
    1     101        Aman       98
    2     102        Geet       95
    3     103        Sahil      87
    4     104        Pawan      79
```

Write a program to swap two numbers without using a temporary variable.

```
#Program to swap two numbers
num1 = int(input('Enter the first number: '))
num2 = int(input('Enter the second number: '))
print("\nNumbers before swapping:")
print("First Number:",num1)
print("Second Number:",num2)
(num1,num2) = (num2,num1)
print("\nNumbers after swapping:")
print("First Number:",num1)
print("Second Number:",num2)
```

Output:

```
Enter the first number: 5
Enter the second number: 10
Numbers before swapping:
First Number: 5
Second Number: 10
Numbers after swapping:
First Number: 10
Second Number: 5
```

Write a program to compute the area and circumference of a circle using a function.

```
def circle(r):          # Function to compute area and circumference of the circle
area = 3.14*r*r
circumference = 2*3.14*r          # returns a tuple having two elements area and
                                    circumference
return (area, circumference)      # end of function

radius = int(input('Enter radius of circle: '))
area, circumference = circle(radius)
print('Area of circle is:', area)
print('Circumference of circle is : ', circumference)
```

Output:

```
Enter radius of circle: 5
Area of circle is: 78.5
Circumference of circle is: 31.400000000000002
```

Print the maximum and minimum number from this tuple.

```
numbers = tuple()                                   #create an empty tuple 'numbers'
n = int(input("How many numbers you want to enter?: "))
for i in range(0,n):
        num = int(input())      # it will assign numbers entered by user to tuple
                                  'numbers’
        numbers = numbers +(num,)
print('\nThe numbers in the tuple are:')
print(numbers)
print("\nThe maximum number is:")
print(max(numbers))
print("The minimum number is:")
print(min(numbers))
```

Output:

```
How many numbers do you want to enter? : 5
 9  8  10 12 15
The numbers in the tuple are: (9, 8, 10, 12, 15)
The maximum number is : 15
The minimum number is : 8
```